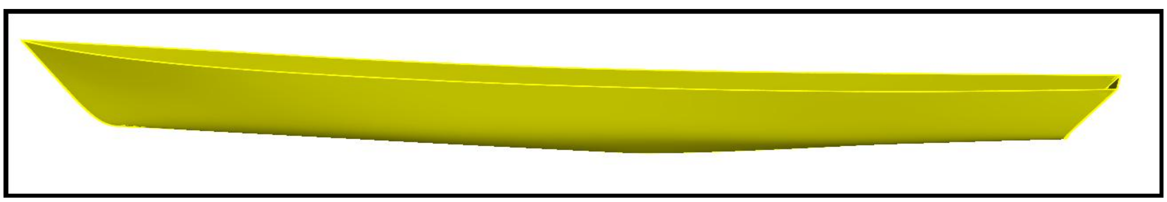
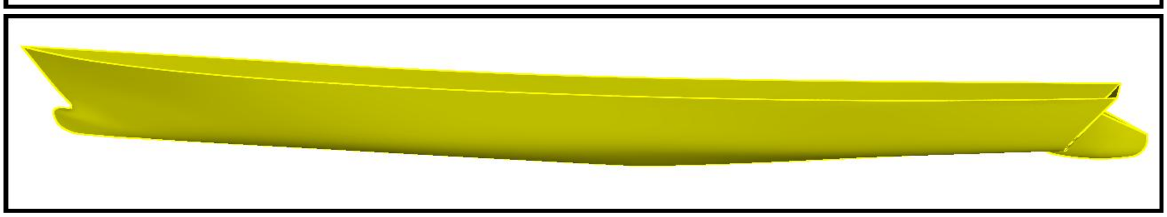
Naval Surface Warfare Center
Carderock Division
West Bethesda, MD 20817- 5700

NSWCCD-CISD–2012/004 July 2012
Center for Innovation in Ship Design
Summer Intern Report

Hull Form Design and Optimization Tool Development

By:
Lacy Thomas
Brendan White





Approved for Public Release: Distribution is unlimited

NSWCCD-CISD-2012/004 / July 2012 / Hull Form Design and Optimization Tool Development

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE	**2. REPORT TYPE**	**3. DATES COVERED** *(From-To)*
September 7 2012	Final	May 2012 - August 2012

4. TITLE AND SUBTITLE
Hull Form Design and Optimization Tool Development

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Brendan White
Lacy Thomas

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) AND ADDRESS(ES)
Naval Surface Warfare Center
Carderock Division
9500 Macarthur Boulevard
West Bethesda, MD 20817-5700

8. PERFORMING ORGANIZATION REPORT NUMBER
NSWCCD-CISD-2012/004

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Chief of Naval Research
One Liberty Center
875 North Randolph Street, Suite 1425
Arlington, VA 22203-1995

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for Public Release: Distribution Unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
George Mason University's (GMU) Center for Computational Fluid Dynamics (CCFD) is working with the Naval Surface Warfare Center's (NSWC) Center for Innovation in Ship Design (CISD) under a five year grant from the Office of Naval Research (ONR) aimed at developing and validating hull form design and hydrodynamic optimization tools. These tools will be used for early stage ship design and education of junior naval engineers. The categories of the tools can be further broken down into tools for hull form generation, representation and modification, design-oriented simple Computational Fluid Dynamic (CFD) tools, and hull form optimization tools. This paper will discuss the further development of these tools including the implementation and testing of a new optimization algorithm, the improvement of a rapid hull form generation tool (HulGen), and the creation of a user interface for a SWATH resistance and powering prediction program (SWAD). The work was performed under the 2012 Naval Research Enterprise Intern Program.

15. SUBJECT TERMS
hydrodynamic, hull form, generation, optimization, algorithm

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Colen Kennell
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	SAR	34	**19b. TELEPHONE NUMBER**
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED			301-227-5468

Team

Brendan White

Lacy Thomas

Acknowledgements

Chi Yang - GMU
Max Harper - CISD
Bob Lamb - CISD
Colen Kennell - CISD
Jack Offutt – CISD
Steven Ouimette–CISD
Siu Fung – NSWCCD
Matthew Knobloch - SEAP
Center for Computational Fluid Dynamics – GMU

Table of Contents

List of Figures

List of Tables

Abbreviations

CISD – Center for Innovation in Ship Design
CFD – Computational Fluid Dynamics
DE – Differential Evolution
DWL – Design Waterline
EHP – Effective Horsepower
GMU – George Mason University
NREIP – Naval Research Enterprise Intern Program
NSWCCD – Naval Surface Warfare Center Carderock Division
NURBS – Non Uniform Rational B-Spline
ONR – Office of Naval Research
SSF – Steady Ship Flow
SWATH – Small Waterplane Area Twin Hull
TriSWACH – Trimaran Small Waterplane Area Center Hull
UI – User Interface
UNREP – Underway Replenishment

Abstract

George Mason University's (GMU) Center for Computational Fluid Dynamics (CCFD) is working with the Naval Surface Warfare Center's (NSWC) Center for Innovation in Ship Design (CISD) under a five year grant from the Office of Naval Research (ONR) aimed at developing and validating hull form design and hydrodynamic optimization tools. These tools will be used for early stage ship design and education of junior naval engineers. The categories of the tools can be further broken down into tools for hull form generation, representation and modification, design-oriented simple Computational Fluid Dynamic (CFD) tools, and hull form optimization tools. This paper will discuss the further development of these tools including the implementation and testing of a new optimization algorithm, the improvement of a rapid hull form generation tool (HulGen), and the creation of a user interface for a SWATH resistance and powering prediction program (SWAD). The work was performed under the 2012 Naval Research Enterprise Intern Program.

Introduction

Background and Objectives

Hydrodynamic design of ships involves several stages, from early-stage and preliminary design to final design. It has become increasingly important to evaluate multiple hydrodynamic aspects of performance simultaneously and efficiently during the early stage of the design process.

The Center for Computational Fluid Dynamics (CFD) at George Mason University has an ongoing research project sponsored by the Office of Naval Research (ONR) on the development of hull form design and hydrodynamic optimization tools. These tools are comprised of three main components. One component consists of design-oriented CFD tools. The second consists of hull form generation, representation, and modification tools. The final component consists of optimization tools.

The 2012 Naval Research Enterprise Intern Program (NREIP) intern team was tasked with the further development of the tools. As part of this task the team was to implement and validate an optimization algorithm, improve a rapid hull form generation tool called HulGen, and create a user interface for a SWATH ship total resistance and powering prediction tool, called SWAD.

Deliverables

The summer work consists of three individual sections. Each of these sections has its individual set of associated deliverables.

Hull Optimization Tool Development

- DE input files for analytical functions
- Specified range wave drag calculation tool
- Contour plotting tool
- DE tool for optimizing a TriSWACH
- DE optimization plotting tool
- Basic user guide for contour and DE tools

Hullform Generation Tool Development

- Reduction of required input parameters for HulGen

Resistance and Powering Prediction Tool Development

- New user interface for SWAD
- Graphic feedback and accessible output data from SWAD

Hull Optimization Tool Development

Background

One of the objectives for the summer was to create and validate a hull optimization tool using an existing optimization algorithm. The intent of the optimization tool was to find the optimal position for the sidehull of a TriSWACH, shown in Figure 1, to minimize the wave drag. Scilab, open source software, was selected to create this optimization tool**.** This software is similar to and has many of the same capabilities as the more widely used Matlab, which is not open source. The algorithm to be used for the optimization process was the Differential Evolution (DE) algorithm. This Differential Evolution algorithm is available as open source software. It was developed by Kenneth Price and Rainer Storn from the University of California at Berkley and is available for many different programming environments including a version which can be run in Scilab.

The DE algorithm is used to find the location of the global minimum of a function, not the value of the global minimum. The algorithm accomplishes this by using a method known as metaheuristics which allows the algorithm to examine a large area by trying to iteratively improve an original guess as to where the minimum is located. These iterations are carried out until an output value less than or equal to the user specified minimum is obtained. After the initial solution attempt is generated from bounds specified by the user, the algorithm generates possible solutions by combining previous solution attempts according to some formula within the algorithm and keeping track of the solution attempt with the lowest value.

Objective

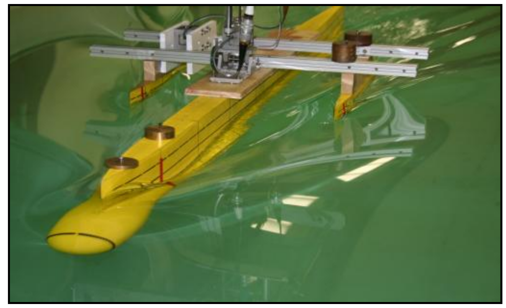


Figure 1: TriSWACH Model

The goal was to integrate the DE algorithm into a hull form optimization tool used to find the location of the sidehull of a TriSWACH with minimum wave drag. A picture of the TriSWACH model can be seen in Figure 1. The algorithm was to be validated before it would be accepted for use in an optimization tool. This was to be done with a variety of analytical functions. Upon the algorithm's validation, a new optimization tool was to be created by integrating the DE algorithm with two already existing hydrodynamic optimization programs developed at George Mason University. These programs include a small geometry generation program and the Steady Ship Flow (SSF) program used for calculating wave drag. Upon completion of this tool, a separate tool was developed to validate the results.

Algorithm Validation

To validate the DE algorithm, input files were written for nine different analytical functions including Rosenbrock's Saddle Function, shown in Figure 2 and the Greiwank Function shown in Figure 3. Two input files are required for each function. One input file contains a computer coded function representing the mathematical function. The other input file contains the parameters for running the function code with the algorithm. These parameters include the number of inputs (dimensions), the minimum value the algorithm is trying to obtain, minimum bounds for the points, maximum bounds for the point, and a maximum number of iterations. It should be noted that the minimum and maximum bounds are not constraints, but rather an initial guess as to where the minimum may be found. This gives the algorithm bounds to generate an initial guess to find the minimum. The algorithm will still search outside these bounds if necessary. If the user wishes to put actual limitations on where the algorithm can search, as seen in Rosenbrock's Saddle Function in Figure 2, constraints can be added to the function file which can force the algorithm to remain strictly within the given bounds.

$$f_2(\underline{x}) = 100 \cdot (x_0^2 - x_1)^2 + (1 - x_0)^2;$$
$$x_j \in [-2.048, 2.048]$$

Figure 2: Rosenbrock's Saddle Function

Only functions for which the global minimum was already known were used for validation. The algorithm searches for the global minimum given by the user and will stop when either a value less than or equal to the specified minimum is found or when the algorithm has reached the maximum number of iterations. The algorithm was able to find the global minimum of each tested function.

Greiwank Function

When testing the DE algorithm, one function was used more than others. This is the Greiwank Function. It has some special properties that made it a beneficial test function for the algorithm. Figure 3 shows the general Greiwank Function. For testing purposes, an “n” value of two was used.

$$F(\mathbf{x}) = \sum_{i=1}^{n} \frac{(x_i)^2}{C_1} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

Figure 3: Greiwank Function

The Greiwank Function has many local minima, which makes the function an interesting test case for it is able to specifically test the algorithm’s ability to find the global minimum without getting stuck in any local minima. Figure 4 shows a contour plot of the function. The local minima are shown by the blue circles tending towards the top right corner where the global minimum is located and plotted on the contour.

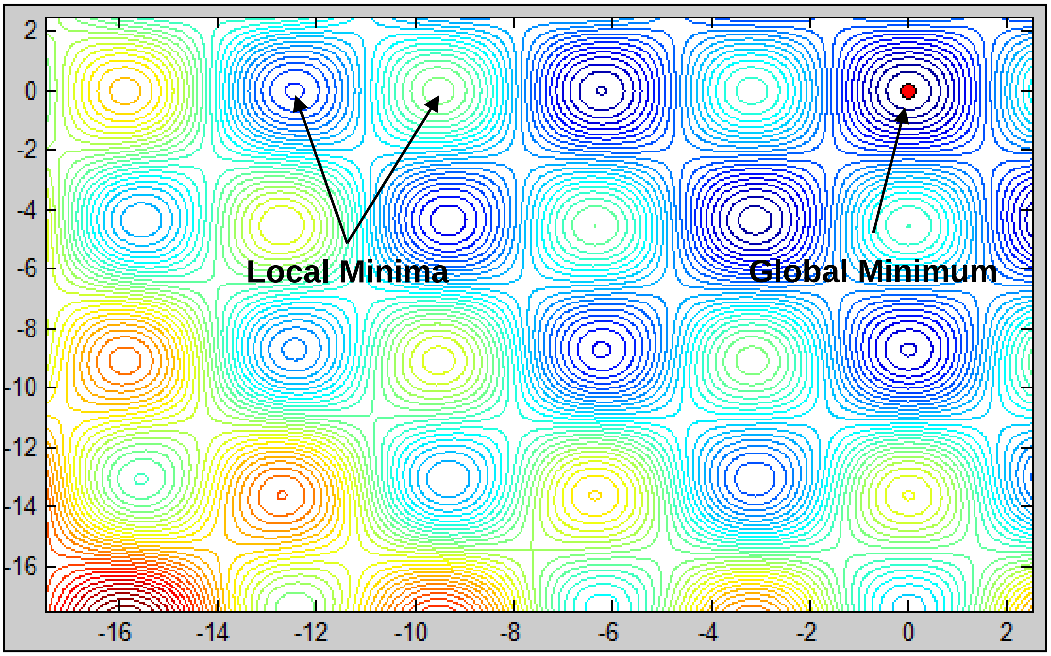


Figure 4: Contour of Greiwank Function

For testing purposes, the program was set to find zero with a precision of six decimal places. This was to ensure a fairly accurate result with a reasonable number of iterations. The program generates a random number between the given bounds for the first attempt at finding the minimum. This causes the DE algorithm to require a different number of iterations for every run because it generates a different path for finding the minimum each time. The algorithm was able to find the minimum very quickly, which is a feature beneficial for the future implementation of the algorithm as a hydrodynamic optimization tool. A sample test result is shown in Figure 5, which presents a convergence history of the function and the location with reference to the number of iterations.

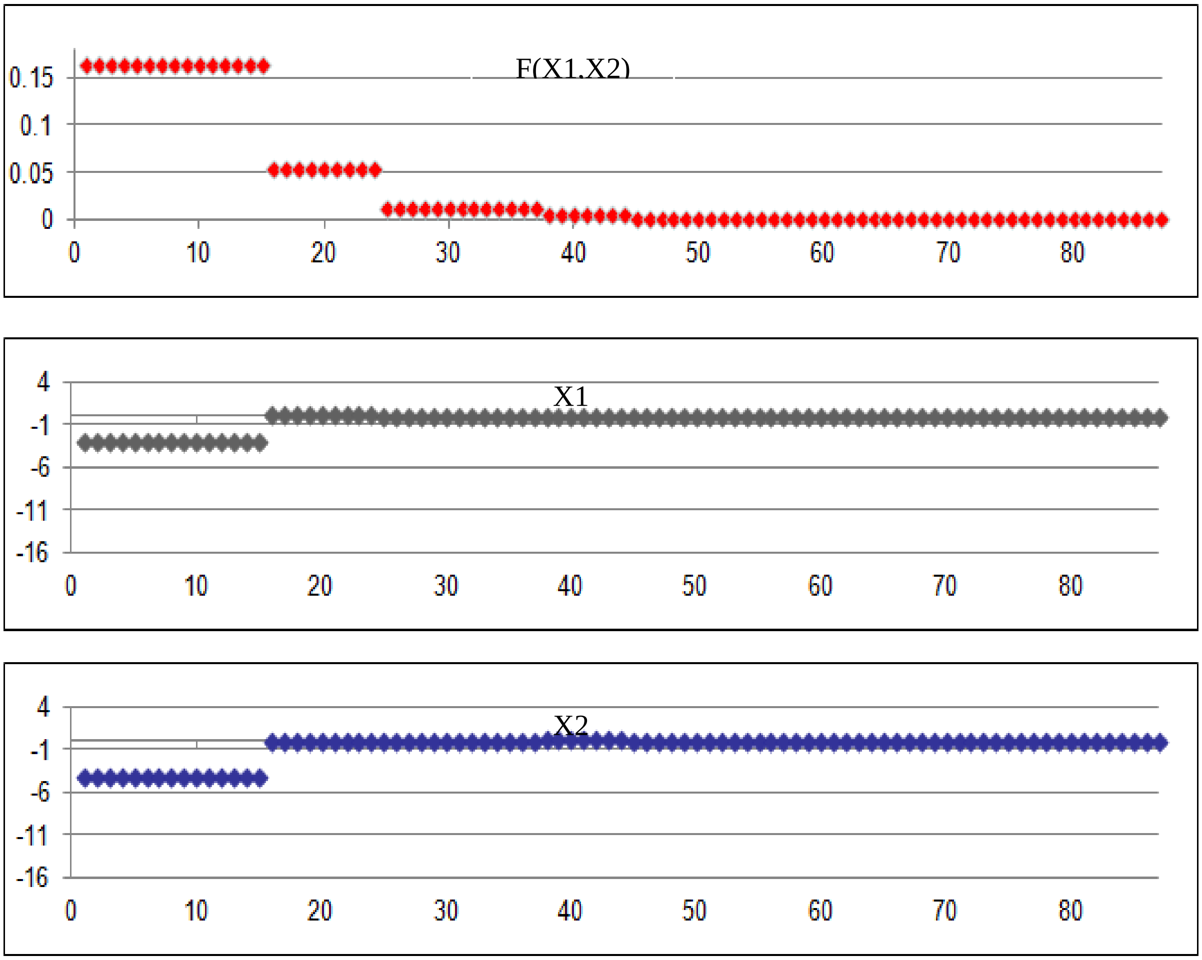


Figure 5: Greiwank Convergence History

The algorithm was successful in finding the global minimum each time it was run. The results provided evidence that the algorithm would be valid for the optimization tool.

Algorithm Implementation

After validating the DE algorithm, the next step was to implement the algorithm as a hydrodynamic optimization tool. Upon completion of this tool, it was to be applied to a TriSWACH model to investigate the effects of longitudinal and transverse spacing of the sidehull on the wave drag of the model, and to find the optimal position of the sidehull for low wave drag. The sidehull spacing definition is shown in Figure 6 which defines the three different longitudinal positions and three different transverse positions considered in the experiment.

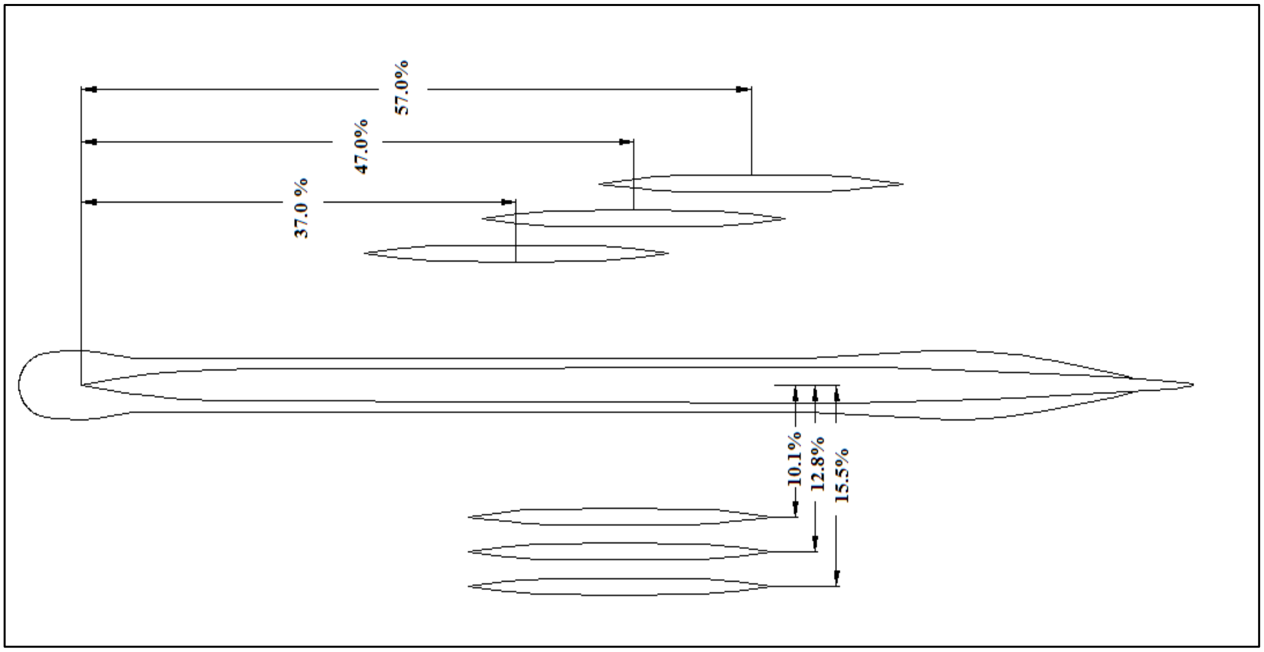


Figure 6: Sidehull Positions for TriSWACH

Single Speed Input Files

To use the DE algorithm for the TriSWACH, the wave drag at each sidehull position had to be written as a function. Two programs from George Mason University (GMU) were used to help accomplish this task. The first was a small geometry generation program that created a geometry file for a TriSWACH given a specific sidehull location. This geometry generation program uses an input file in which the transverse and longitudinal location of the sidehull can be changed and generates new TriSWACH geometry with the given sidehull position. The programs from GMU normalize the ship to a length of one. Since the TriSWACH used for the test was 84 feet long, it was necessary to divide the considered locations by 84 to obtain the proper format used by the GMU programs. This conversion issue was considered when writing the input file and a variable for the length of the ship was included to allow the program to work with different ship lengths. The geometry generation program produces an output file containing the new geometry which is used by the second GMU program, the Steady Ship Flow (SSF) program, which calculates the wave drag for the given geometry. The drag is calculated at a given speed or Froude Number specified by the user in the SSF program's input data file.

The first input file for the optimization tool contains a coded representation of a function with the return value being the value the user is trying to minimize. The tool was being used to minimize drag, thus code had to be written to implement a function with drag as the return value. This was done by coding the wave drag as a function of the longitudinal position and the transverse position of the sidehull. The tool generates longitudinal and transverse position values to be used as input values for this coded function. This code then writes an input file for the geometry generation program with the new position coordinates. The geometry generation program is executed and an output file containing the new ship's geometry is generated. Upon generation of this output file, SSF is called and a wave drag output file is produced. The code then reads this new output file and adds the current drag values to an array containing all drag values from previous iterations. The current drag value is the coded function's return value.

The basic information contained in the optimization tool's parameter file, the second input file, is similar to the information used in the parameter files for the analytical functions, with the only real difference being the minimum value. With the analytical functions, the minimum value is known, so that value can be input and the algorithm will stop once that value is reached. The minimum wave drag is unknown, but there is a simple way to work around this problem. The optimal position of the sidehull is the position with the least amount of drag. Therefore the minimum value is set to zero. Knowing there was not going to be a location where the drag will equal zero, the intent is to have the algorithm find the closest value to zero within the maximum number of iterations.

Multiple Speed Input Files

Once the input files were created and results were obtained using a single speed, the first input file was changed slightly to accommodate multiple speeds. The optimization tool can find the optimal position when a ship is traveling at a specific speed, but it is not always the case that only one ship speed is of interest. This creates the need to incorporate multiple speeds into the tool. The relative importance of each speed is implemented with different weights. If the user wants the optimal position for a ship that runs at one speed seventy-five percent of the time and another speed twenty-five percent of the time, the weights of the numbers can be adjusted. For the summer work, the weights were assumed to be equal across all speeds. This required only a simple average calculation. The additional speed values must also be added to the SSF's input data file before the average calculations can be done.

Optimization Tool Validation

Before the optimization tool's results could be accepted, the results had to be validated. This was done by creating a separate contour tool in Scilab to draw contour plots for wave drag and comparing the minimum found in the contour plot to the minimum found with the DE algorithm.

The longitudinal and transverse points are not generated for the contour tool in the same way they are generated when using the optimization tool. The contour tool has to generate its points autonomously rather than having the points generated by the DE algorithm. The points are generated by using a minimum and maximum variable in both the longitudinal and transverse directions. The user then specifies the number of points desired in each direction and the tool evenly divides the given dimensions into the correct number of points. The tool iterates through the points calculating drag at each possible combination of longitudinal and transverse points and stores these values in an array similar to how the values are stored in the DE implementation. This array is then stored in a text file. There is a separate plotting tool written for the actual plotting to allow the user to plot existing values without having to run the entire contour tool. The plotting tool simply opens the text file where the output is stored and creates a contour plot with the longitudinal points on the x-axis and the transverse points on the y-axis. The contour plot has colors ranging from red for the higher values to dark blue and black for the lowest values. By using the contour plots, the results from the DE algorithm can be validated. The contour plots provide a visual from which the minimum value can easily be estimated. This value can be compared to that which was obtained using the DE algorithm.

Several cases were tested and the contour plot results were compared against the results from the DE algorithm. In Figure 7 the minimum is represented by the circle towards the middle of the contour plot. This contour plot was created using sixty longitudinal points and twenty transverse points for a total of 1,200 points. A Froude Number of .31 was used for these calculations.

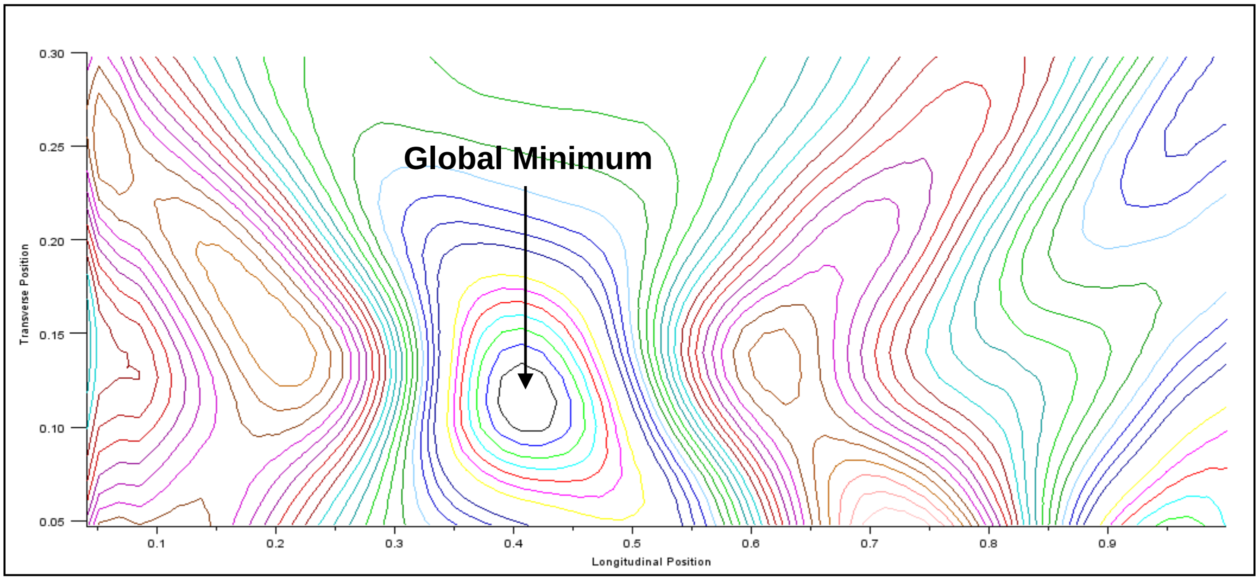


Figure 7: Contour with 1,200 Points

Once the contour plot was created, the results were compared with those from the optimization tool. The results from the tool were consistently close to the minimum after 30 iterations. Trials were run using up to 60 iterations, but it was found that accurate results were achieved by 30 iterations as demonstrated by Figure 8. To obtain accurate results in a reasonable amount of time, the optimization tool was configured with a maximum iteration limit of 30.

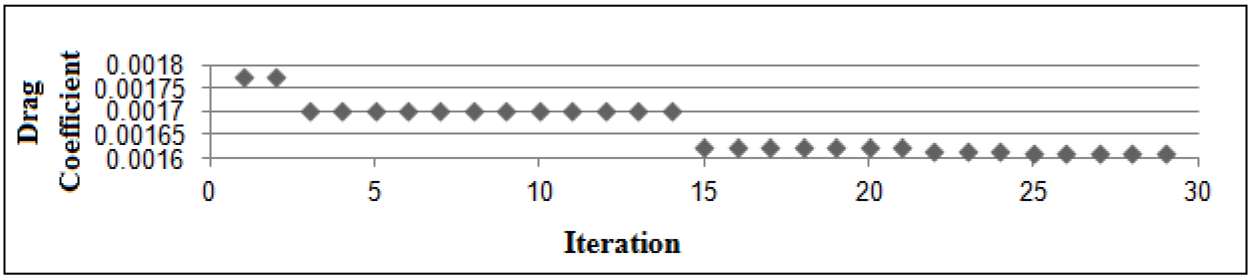


Figure 8: Drag Convergence Using Differential Evolution

Table 1 presents the optimal sidehull positions obtained using the optimization tool for Fn = 0.31, 0.25, and the average Fn, respectively.

As shown in the second row of Table 1, the optimization tool's results for a Froude Number of .31 agree with the results from the contour plot. This demonstrates that the tool was successful in finding the optimal position.

Table 1: DE Results for Optimal Sidehull Location

Case	Longitudinal Position	Transverse Position
Fn = 0.25	0.580 (48.7 ft)	0.052 (4.4 ft)
Fn = 0.31	0.411 (34.5 ft)	0.114 (9.6 ft)
Average Fn	0.943 (79.2 ft)	0.052 (4.4 ft)

Table 2 compares the contour plot results to the DE algorithm results for each of the speeds to further validate the optimization tool.

Table 2: Comparison of Optimal Sidehull Locations from Contour Plot and DE

Longitudinal Position	Contour Plot Range	Differential Evolution
Fn = 0.25	0.56 - 0.61	0.580
Fn = 0.31	0.40 - 0.45	0.411
Average Fn	0.92 - 0.97	0.943
Transverse Position	**Contour Plot Range**	**Differential Evolution**
Fn = 0.25	0.05 - 0.09	0.052
Fn = 0.31	0.09 - 0.13	0.114
Average Fn	0.05 - 0.07	0.052

Figure 9 shows the last iteration of the DE algorithm plotted on the contour plot when a Froude Number of .31 was used. This provides a visual form to show that the results agree. The point representing the last iteration from the DE algorithm is located in the global minimum circle on the contour plot.

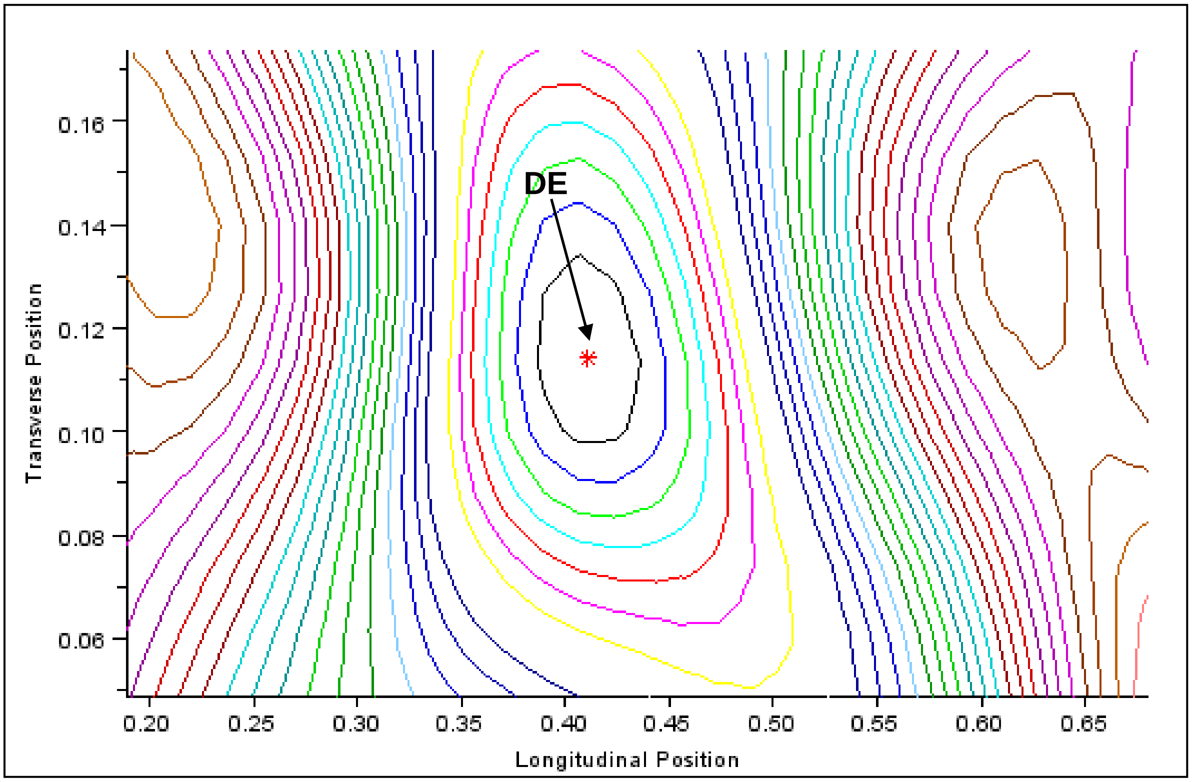


Figure 9: DE Optimal Sidehull Location Plotted on Contour Plot for Froude Number .31

After finding the optimal positions at different speeds using the optimization tool, those positions were examined over a range of speeds. Using the previously mentioned SSF program from GMU, the wave drag for three optimal positions was calculated over a wide range of speeds. Fifty Froude Numbers between 0.18 and 0.48 were used for these calculations. Figure 10 shows the calculated wave drag for the geometry created when the sidehull is in the optimized location determined by the Froude Numbers shown in the legend.

It can be observed from Figure 10 that the optimization performed for a single design speed will result in a large drag reduction at the given speed, but might have a drag increase at off-design speeds. It would be beneficial to consider multi-design-speed optimization.

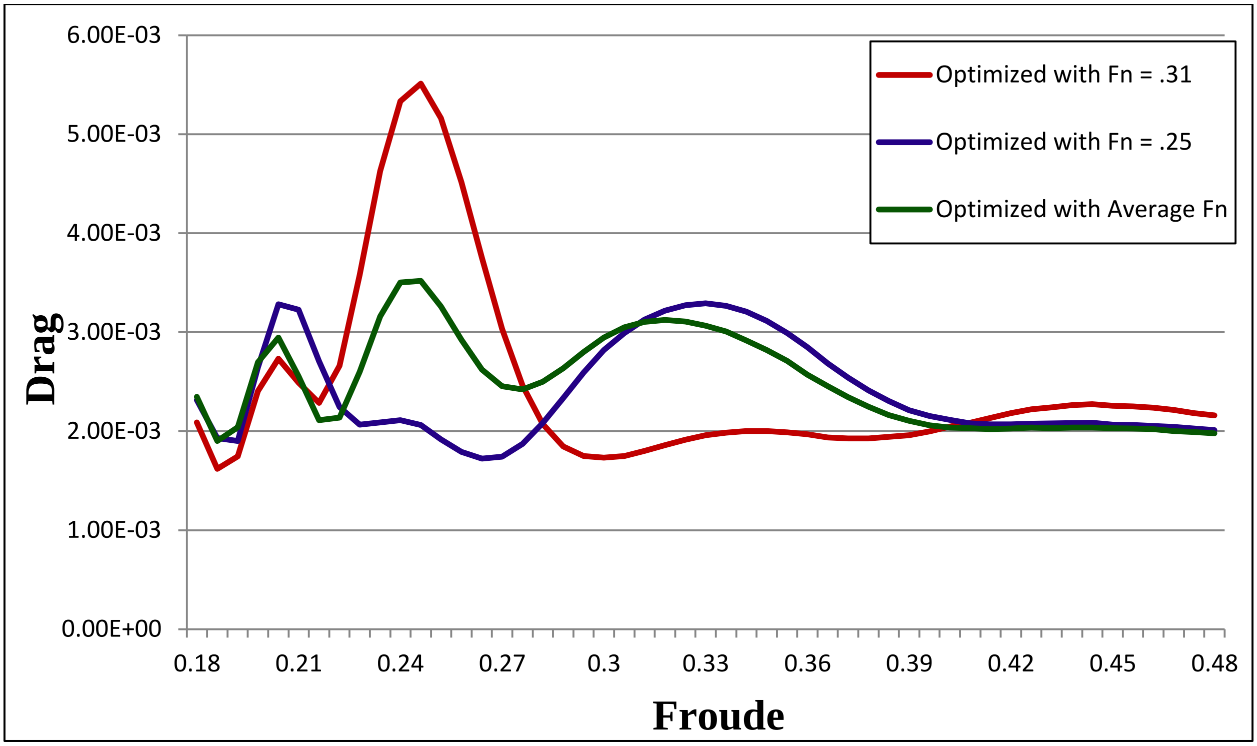


Figure 10: Wave Drag at Optimized Positions

Strengths of the Optimization Tool

Run Time

There are multiple benefits to using the algorithm to find the minimum rather than the contour plots. The optimization tool results provide a specific number as opposed to a range. The run time for the algorithm is also significantly shorter than the runtime of the contour plots. Computing the previous contour plot requires approximately twelve hours on a laptop with a dual-core processor, but the DE optimization tool with thirty iterations requires only three hours on the same hardware. This represents a significant time difference even with the small amount of data that was used. As the size of the data increases, the run time benefits of using the DE optimization tool will increase.

Alternate Applications

Figure 11: SWATH ship

After use of the DE algorithm for optimizing the TriSWACH had been validated, other uses for the algorithm were considered. The DE algorithm is a general optimization algorithm and therefore it can be implemented with a variety of different programs. Again attempting to use the algorithm to optimize drag, the algorithm was applied to SWAD90, which is a power and resistance prediction tool for SWATH (Small Waterplane Area Twin Hull) ships. An example of a SWATH ship is shown in Figure 11. The input files required to use the algorithm with SWAD90 are similar to those for the TriSWACH optimization. When attempting optimization for the TriSWACH, the position of the sidehull was variable in the equation. When optimizing with SWAD90, the variable to optimize was unknown. The first attempts to optimize with SWAD90 involved trying to optimize the nose and midbody lengths of the strut to minimize drag, but after several trial runs the algorithm seemed to have trouble finding a minimum. The results from both the algorithm and the contour plots were inconclusive. This was interpreted to mean that the drag value predicted by SWAD90 is not sensitive to the nose and midbody lengths of the strut. Due to time constraints, experimenting with other variables was left as a future project.

Summary

The Differential Evolution algorithm was implemented into a new hydrodynamic optimization tool. The algorithm was thoroughly tested with several different analytical functions before the development of the optimization tool began. Once the DE algorithm was validated, the optimization tool was implemented. A contour tool was developed to create contour plots to show the wave drag of the sidehull of a TriSWACH in various locations. The contour tool calculates wave drag at a specified number of points between given bounds in the transverse and longitudinal directions. It then creates a contour plot of these points. The contour plots were used to find the optimal position of the sidehull of a TriSWACH in terms of the wave drag. The plots created by this tool were used to help validate the results of the new optimization tool by comparing the minimum found by the optimization tool to the minimum in the contour plots. Finally, the optimization tool was integrated with SWAD90 to explore the tool's versatility and confirm that the tool could be used for other applications.

Hullform Generation Tool Development

Background and Objectives

During the early stages of the ship design process, it is important to be able to evaluate multiple hydrodynamic aspects of performance simultaneously and efficiently. The Center for Computational Fluid Dynamics at George Mason University has an ongoing research project sponsored by the Office of Naval Research on the development of a methodology for innovative hydrodynamic ship hull design.

Rapid ship hullform generation tools are useful in early stage ship design and are a precursor to hydrodynamic performance analysis and design optimization. HulGen is a rapid hull generation tool that generates a hull from scratch using user input hull form parameters. The latest version of HulGen was developed at CISD in 2011. The function of HulGen is to facilitate the generation and modification of monohull forms during early stage ship design.

A new feature has been implemented in HulGen to allow the user to generate a hull form using fewer parameters than was previously required. Use of the full array of HulGen parameters makes it suitable for integration with the design and optimization tools being developed at GMU. The option to use fewer parameters makes HulGen more suitable for use by a human user.

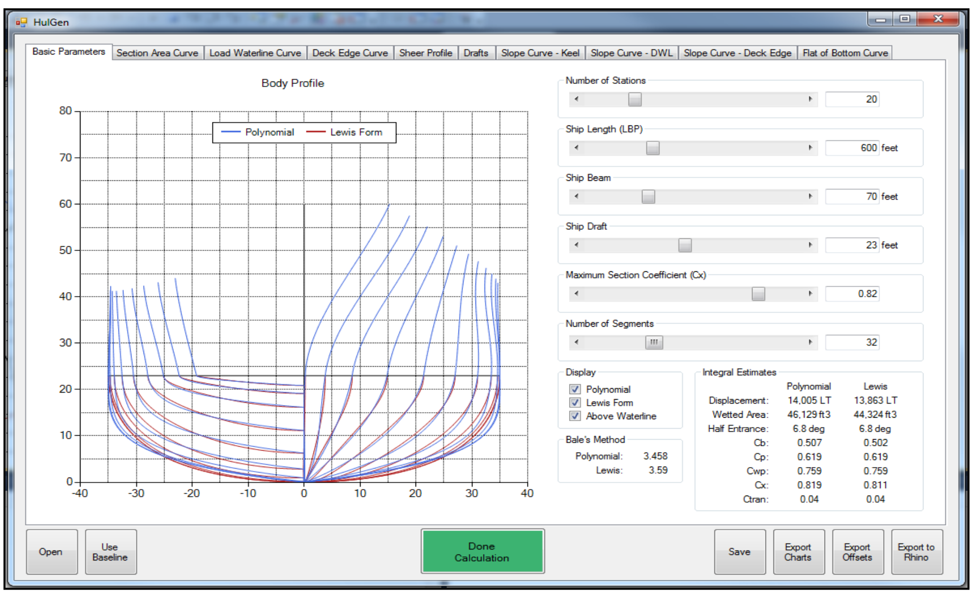


Figure 12: HulGen Main Interface

Existing HulGen Tool

Input

HulGen is implemented in Visual Basic to provide a user friendly interface. The main interface is shown in Figure 12. The hull lines are polynomials defined by hull form parameters input by the user. HulGen uses a total of 68 parameters to generate a hull. The list of parameter values used in HulGen can be seen in Table 3. These parameters are used to create the curves in Table 4 which are then used to create the body plan of the hull.

Table 3: HulGen Parameters

Parameter #	Parameter name
1	Number of Stations
2	Ship Length (LBP)
3	Ship Beam
4	Ship Draft
5	Maximum Section Coefficient (Cx)
6	Number of Segments
7	Area at FP (SAC)
8	Slope of Curve at FP (SAC)
9	Location of Desired Station of Maximum Area (Xmax) (SAC)
10	Slope of Curve at Xmax (SAC)
11	Length of Parallel Sections (Lmid) (SAC)
12	Area at AP (Ctran)
13	Slope of Curve at AP
14	Prismatic Coefficient (Cp)
15	Longitudinal Center of Buoyancy
16	Offset at FP (LWC)
17	Slope of Curve at FP (LWC)
18	Location of Desired Station of Maximum Area (Xmax) (LWC)
19	Slope of Curve at Xmax (LWC)
20	Length of Parallel Sections (Lmid) (LWC)
21	Offset at AP (LWC)
22	Slope of Curve at AP
23	Waterplane Coefficient (Cwp)
24	Longitudinal Center of Floatation (LCF)
25	Offset at FP (FDEC)
26	Slope of Curve at FP (FDEC)

27	Location of Flat Middle Section (Xmid)
28	Length of Flat Middle Sections (Lmid)
29	Offset at Forward End of Flat Middle Section (Ymid)
30	Slope of Curve at Forward End of Flat Middle Section (Smid)
31	Offset at AP
32	Slope of Curve at AP
33	Depth at FP
34	Depth at Midships
35	Depth at AP
36	Longitudinal Position of Keel Rise
37	Section Area Coefficient at AP
38	Slope Value at FP (Keel)
39	Slope Rate at FP (Keel)
40	Location of Flat Middle Section (Xmid) (Keel)
41	Length of Flat Middle Sections (Lmid) (Keel)
42	Slope Value at Forward End of Flat Middle Sections (Keel)
43	Slope Rate at Forward End of Flat Middle Sections (Keel)
44	Slope Value at AP (Keel)
45	Slope Rate at AP (Keel)
46	Slope Value at FP (DWL)
47	Slope Rate at FP (DWL)
48	Location of Flat Middle Section (Xmid) (DWL)
49	Length of Flat Middle Sections (Lmid) (DWL)
50	Slope Value at Forward End of Flat Middle Sections (DWL)
51	Slope Rate at Forward End of Flat Middle Sections (DWL)
52	Slope Value at AP (DWL)
53	Slope Rate at AP (DWL)
54	Slope Value at FP (Deck Edge)
55	Slope Rate at FP (Deck Edge)
56	Location of Flat Middle Section (Xmid) (Deck Edge)
57	Length of Flat Middle Sections (Lmid) (Deck Edge)
58	Slope Value at Forward End of Flat Middle Sections (Deck Edge)
59	Slope Rate at Forward End of Flat Middle Sections (Deck Edge)
60	Slope Value at AP (Deck Edge)
61	Slope Rate at AP (Deck Edge)
62	Longitudinal Location of the Start of FBC (X1)
63	Longitudinal Location of the End of Full Section (X2)
64	Longitudinal Location of the Start of Full Section (X3)

65	Longitudinal Location of the End of FBC (X4)
66	Slope Value at the Start of FBC (S1)
67	Slope Value at the End of FBC (S4)
68	Half Siding Coefficient

Table 4: HulGen Parameter Categories

Category Name	Parameter #s
Basic Parameters	1 to 6
Section Area Curve	7 to 15
Load Waterline Curve	16 to 24
Deck Edge Curve	25 to 32
Sheer Profile	33 to 35
Keel Rise Curve	36 to 37
Slope Curve (Keel)	38 to 45
Slope Curve (DWL)	46 to 53
Slope Curve (Deck Edge)	54 to 61
Flat of Bottom Curve	62 to 68

Each curve is defined by a polynomial with a specific order. The polynomials are calculated in a non-dimensional coordinate system and translated afterwards. The coordinates of the polynomials are normalized using different hull parameters.

Output

HulGen has the ability to create a NURBS surface in Rhinoceros, a commercial 3D modeling software widely used at the Naval Surface Warfare Center. HulGen is also able to export hull offsets and diagrams.

Reduction of Mandatory Input Parameters

HulGen has 68 input parameters used to generate a hull form. However, using 68 different parameters to create a hull form from scratch can be a difficult task for an inexperienced user. A new feature has been implemented in HulGen which provides the ability to approximate 62 parameters based on the other 6 parameters. The parameters are approximated using modified versions of formulas provided by a senior Naval Architect at the Naval Surface Warfare Center with extensive experience in hull design. The formulas used are dependent on the type of ship. The 6 main parameters which are used to calculate the remaining 62 can be found in Table 5.

Table 5: HulGen Main Parameters

Parameter #	Parameter Name
2	Ship Length
3	Ship Beam
4	Ship Draft
5	Maximum Section Coefficient
14	Prismatic Coefficient
34	Depth at Midships

The 6 main parameters can easily be adjusted within the user interface. The algorithm for determining each of the 62 remaining parameters is unique to the parameter and dependent on the type of ship. The user must specify the ship type using a drop down menu. The interface for this feature can be seen in Figure 13.

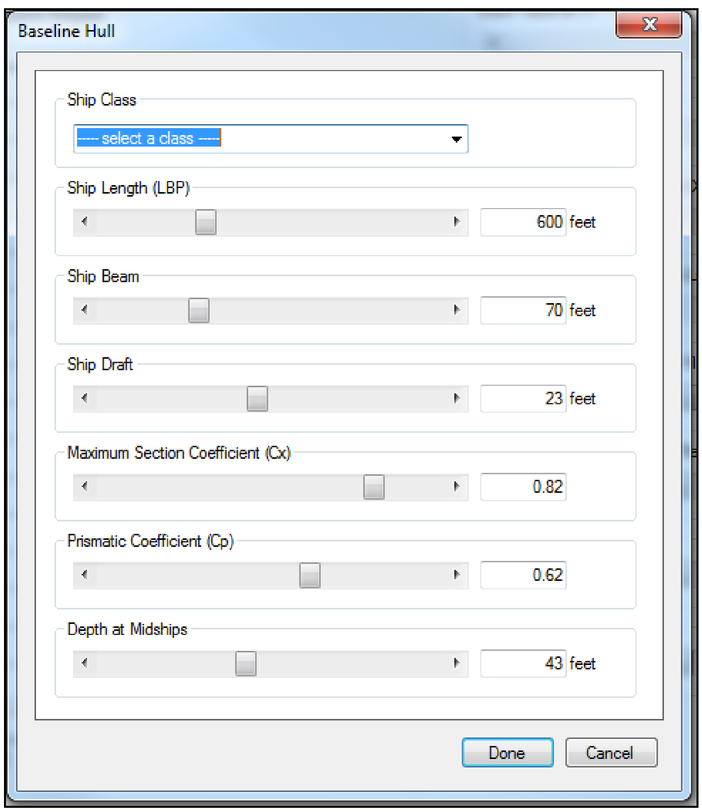


Figure 13: Baseline Hull Interface

The input reduction formulas implemented within HulGen, Figure 14, support the following types of ships:

- Conventional Surface Combatants
- Unconventional Surface Combatants
- Twin-Screw, Open-Stern, UNREP and Auxiliary
- Single-Screw, Closed-Stern, UNREP and Auxiliary
- Single-Screw, Open-Stern, UNREP and Auxiliary
- Aircraft Carriers

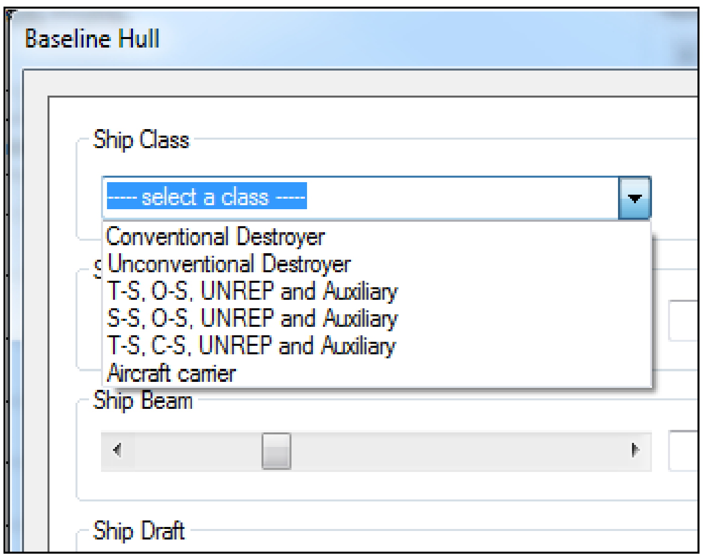


Figure 14: Baseline Ship Type Selection Menu

This new feature allows a user to generate a hull by choosing a hull type and providing 6 main parameters. HulGen will then fill all of the other parameters in for the user. The 6 provided parameters and the 62 approximated parameters can then be used as a design baseline. The user still has the ability to manipulate all 68 parameters before or after using the new feature, thereby allowing the user to create hull forms that do not lie within the baseline types of ships.

Summary

A new feature has been implemented in HulGen which will allow an inexperienced user to generate, modify, and export a hull form using only 6 parameters. The user is still able to use the other HulGen features to modify hull forms to a greater degree. HulGen is suitable for future integration with the CFD tools under development at George Mason University.

Resistance and Powering Prediction Tool Development

Background and Objectives

SWAD is a software tool developed and used by the Naval Surface Warfare Center which predicts total resistance and powering in calm water for Small Waterplane Area Twin Hull (SWATH) ships. Several versions of SWAD exist. The most commonly used version is SWAD90, which was written in FORTRAN and is run using text files and command line inputs. The most recent previous implementation was at CISD as an Excel tool which lacks a detailed graphic user interface. The goal of this project was to provide SWAD an intuitive graphic user interface as a stand-alone application. This was done by implementing SWAD in Visual Basic, creating the newest version SWAD12. The user interface of SWAD12 can be seen in Figure 15.

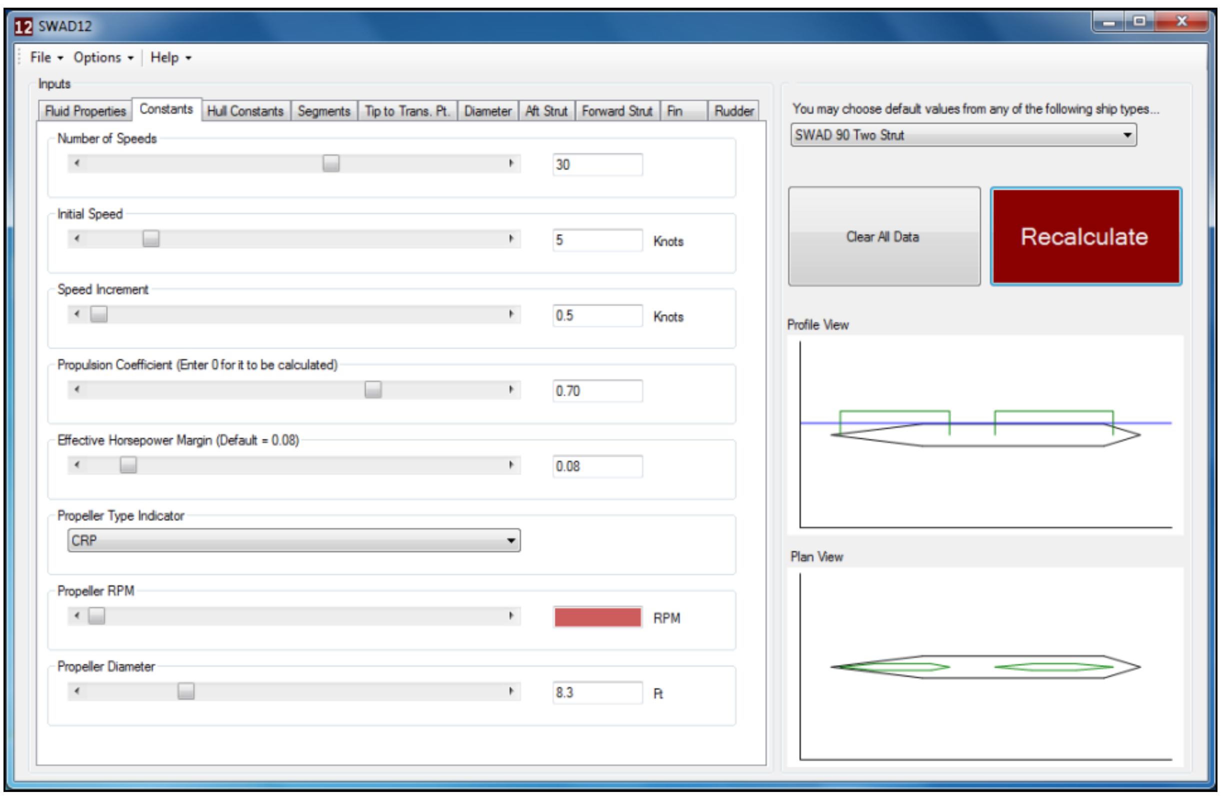


Figure 15: SWAD12 Main Interface

SWAD12 Input

An interface similar to that of HulGen has been implemented to provide an intuitive layout. The input parameters are divided among labeled tabs which the user can easily navigate through. Example hulls have been included in the program, which may serve as a baseline for the user to modify. Parameters that have a limited quantity of discrete possible values are represented as dropdown menus. All other parameters may be easily adjusted through the use of slider bars or by entering exact parameter values into textboxes. One tab of parameters including a dropdown menu and several slider bars is shown in Figure 16. Any change to the value of either the slider bar or the textbox automatically triggers an update to the other. Each slider bar has an associated function which translates from the slider bar's range of integer values to the parameter's range of valid values. Each slider bar's function was designed to provide the slider bar with a reasonable resolution as well as an appropriate minimum and maximum. Most parameters require a linear scaling function, but some parameters use an exponential scaling function. For example, when manipulating the slider bar for ship length, the parameter value will increase faster as the slider bar moves towards the larger values. In previous versions of SWAD, all parameter values were manually entered into a text file in a specific format and order before running SWAD and could not be altered as easily. The dropdown menus and slider bars are useful when quickly adjusting parameter values.

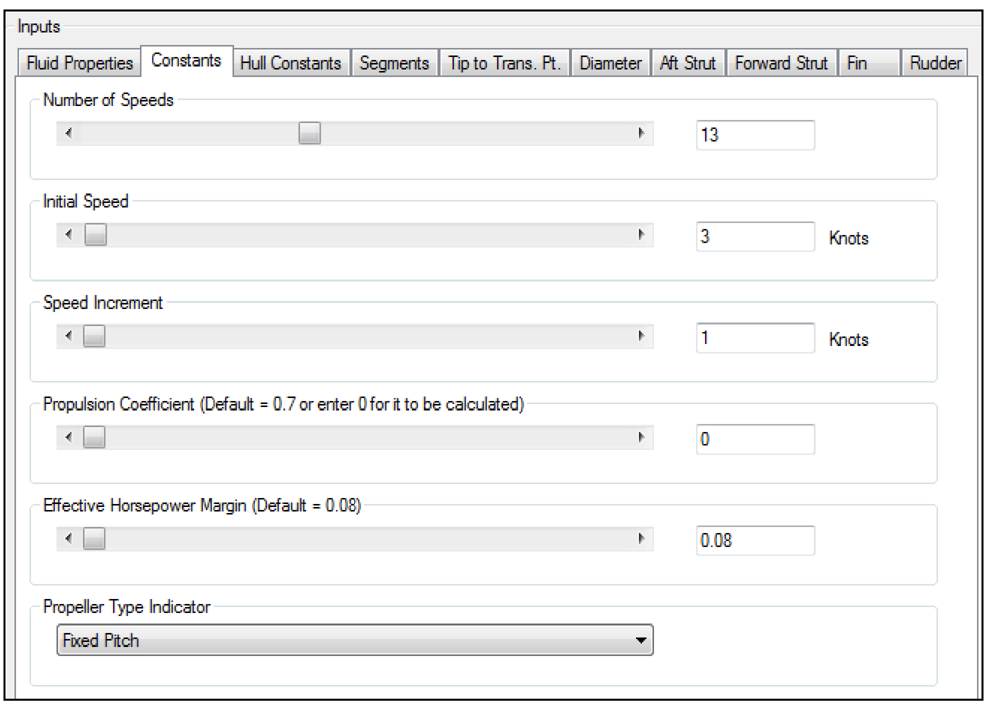


Figure 16: SWAD12 Input Interface

A key feature in the functionality of SWAD12 is backwards compatibility with previous versions of SWAD. SWAD12 was implemented to allow the user to import original SWAD input files compatible with the FORTRAN versions. This import feature is located in the menu bar as shown in Figure 17.

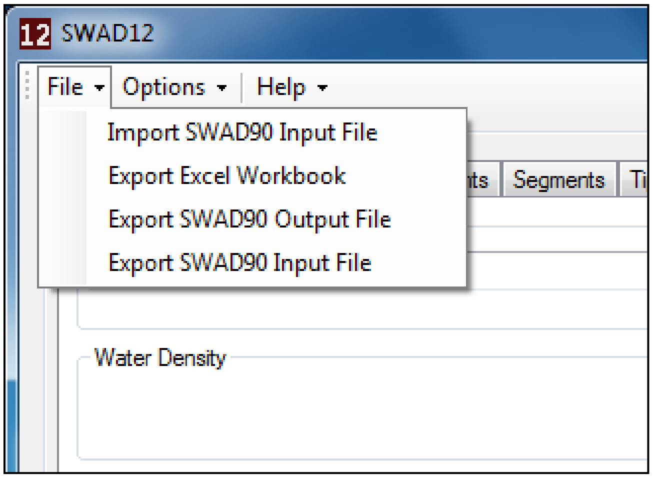


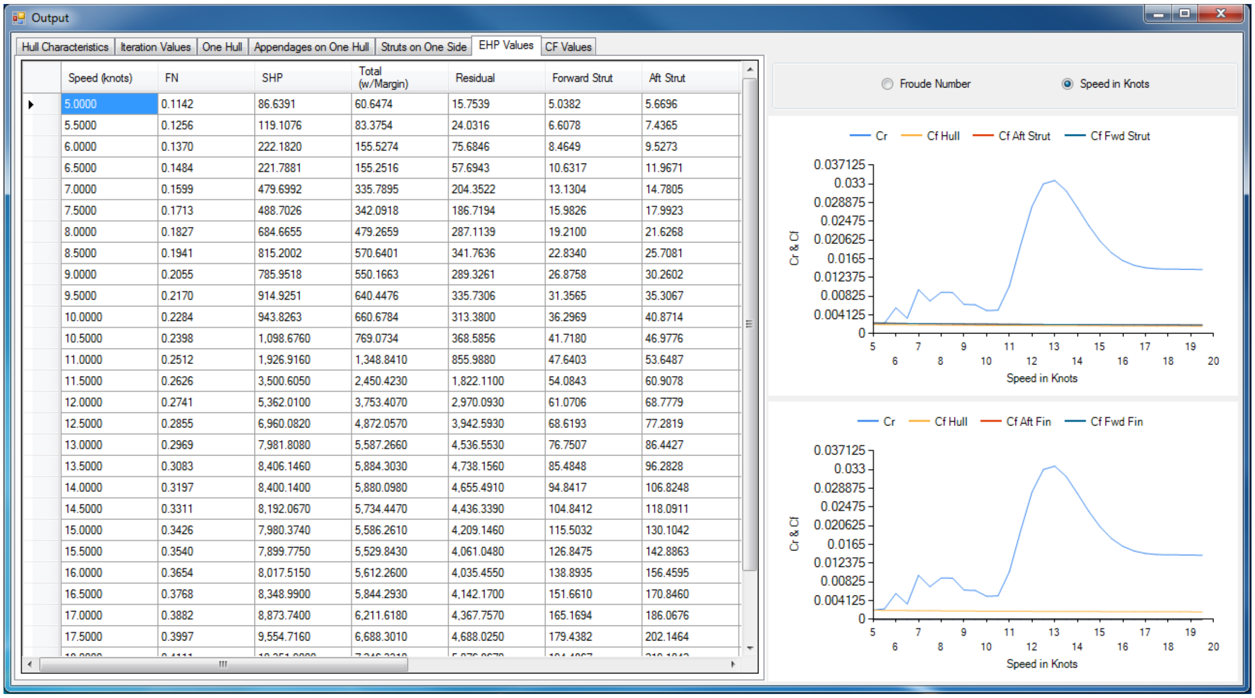
Figure 17: SWAD12 Import and Export Menu

SWAD12 Output

The output interface for SWAD12 can be seen in Figure 18. SWAD12 is concerned primarily with the portions of the ship below the Design Waterline (DWL). Portions above the waterline are included for completeness and because they may be useful when moving the data to another tool.

SWAD12 estimates the resistance and powering of a user specified SWATH hull form for a given set of speeds. Coefficients of residuary resistance, coefficients of frictional resistance, and the effective horsepower are calculated for hulls and struts. SWAD12 also calculates displaced volume and wetted surface area for hulls, struts, and appendages. All of the output data is displayed in Imperial units.

The user has the option to export import data compatible with previous versions of SWAD. The user can export the output data in the same format as FORTRAN versions of SWAD. This feature may useful in the event that individuals have designed scripts to read in data from previous SWAD versions. The user can also export the output data to Excel in the same format as the Excel implementation of SWAD. These options can be seen in Figure 17.



Speed (knots)	FN	SHP	Total (w/Margin)	Residual	Forward Strut	Aft Strut
5.0000	0.1142	86.6391	60.6474	15.7539	5.0382	5.6696
5.5000	0.1256	119.1076	83.3754	24.0316	6.6078	7.4365
6.0000	0.1370	222.1820	155.5274	75.6846	8.4649	9.5273
6.5000	0.1484	221.7881	155.2516	57.6943	10.6317	11.9671
7.0000	0.1599	479.6992	335.7895	204.3522	13.1304	14.7805
7.5000	0.1713	488.7026	342.0918	186.7194	15.9826	17.9923
8.0000	0.1827	684.6655	479.2659	287.1139	19.2100	21.6268
8.5000	0.1941	815.2002	570.6401	341.7636	22.8340	25.7081
9.0000	0.2055	785.9518	550.1663	289.3261	26.8758	30.2602
9.5000	0.2170	914.9251	640.4476	335.7306	31.3565	35.3067
10.0000	0.2284	943.8263	660.6784	313.3800	36.2969	40.8714
10.5000	0.2398	1,098.6760	769.0734	368.5856	41.7180	46.9776
11.0000	0.2512	1,926.9160	1,348.8410	855.9880	47.6403	53.6487
11.5000	0.2626	3,500.6050	2,450.4230	1,822.1100	54.0843	60.9078
12.0000	0.2741	5,362.0100	3,753.4070	2,970.0930	61.0706	68.7779
12.5000	0.2855	6,960.0820	4,872.0570	3,942.5930	68.6193	77.2819
13.0000	0.2969	7,981.8080	5,587.2660	4,536.5530	76.7507	86.4427
13.5000	0.3083	8,406.1460	5,884.3030	4,738.1560	85.4848	96.2828
14.0000	0.3197	8,400.1400	5,880.0980	4,655.4910	94.8417	106.8248
14.5000	0.3311	8,192.0670	5,734.4470	4,436.3390	104.8412	118.0911
15.0000	0.3426	7,980.3740	5,586.2610	4,209.1460	115.5032	130.1042
15.5000	0.3540	7,899.7750	5,529.8430	4,061.0480	126.8475	142.8863
16.0000	0.3654	8,017.5150	5,612.2600	4,035.4550	138.8935	156.4595
16.5000	0.3768	8,348.9900	5,844.2930	4,142.1700	151.6610	170.8460
17.0000	0.3882	8,873.7400	6,211.6180	4,367.7570	165.1694	186.0676
17.5000	0.3997	9,554.7160	6,688.3010	4,688.0250	179.4382	202.1464

Figure 18: SWAD12 Output Interface

Hull Representation

SWAD12 will model twin hulls or a single demi-hull. It allows one strut per hull or two struts per hull. SWAD12 can model simple or bulged hulls with either circular or elliptical cross sections. Hulls are modeled as an elliptical nose cone and a parabolic tail cone with a variable number of cylindrical or conical sections between. The strut waterplane shape is modeled as an elliptical or parabolic nose section, a rectangular parallel mid-section, and a parabolic tail section. Struts are assumed to be wall sided with vertical leading and trailing edges.

Graphic Feedback

Previous versions of SWAD lacked graphic feedback on the user's entered information. With the new interface for SWAD12, valuable feedback is now presented to the user. Coefficients of resistance curves are calculated and graphed with the output. An example can be seen in Figure 19. The user can specify whether the curves should be graphed as a function of the Froude Number or the speed in knots. This is significant as in previous versions of SWAD the user must manually graph the performance curves from the raw output data after exiting SWAD.

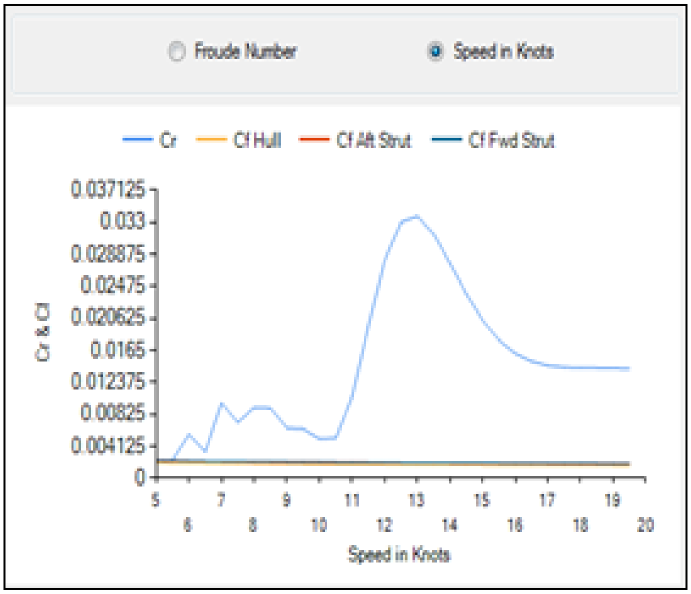


Figure 19: SWAD12 Coefficients of Resistance Curves

The hull corresponding to the input parameters is graphed in the main interface to provide feedback on the dimensions of the hull (Figure 20). The hull is approximated with straight lines and is not meant to represent the curves of the hull used for calculation. The graphed hull is useful, allowing the user to directly see if the information entered is reasonable. In addition, pop up windows have been implemented to warn the user when approximations are made to the hull used for calculations.

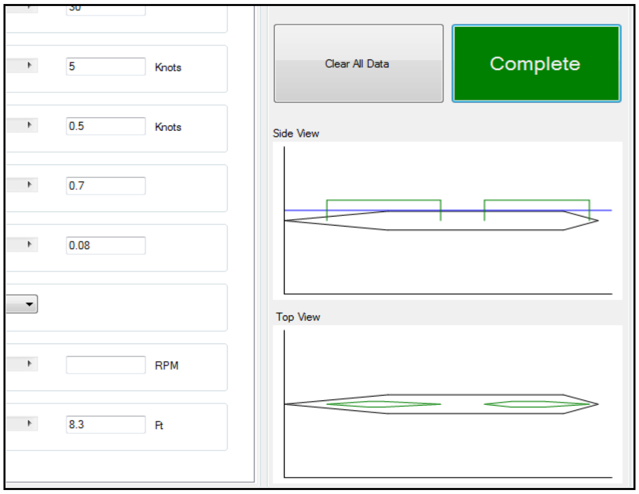


Figure 20: SWAD12 Graph of Hull

Summary

A new tool, SWAD12, has been implemented in Visual Basic. SWAD12 is a SWATH resistance and powering prediction tool which uses a graphic user interface to facilitate user input and to provide graphic feedback. The outputs include plan and profile views of the SWATH, geometry, resistance curves, and EHP data. SWAD12 is a remake of an old FORTRAN SWAD tool. Features have been implemented which will allow backwards compatibility with FORTRAN versions of SWAD.

Recommendations for Further Work

Further development is recommended for all of the tools.

The DE tool currently accommodates up to three speeds, but the tool can be further developed to increase the number of input speeds. This will be useful when trying to optimize ships with missions requiring several speeds. The DE tool can also be linked with HulGen and SWAD12 for further optimization studies.

Improvements to HulGen can expand its ship generation capabilities. Improvements include the ability to model bulbous bows and hard chines. Also, validation of the hull forms generated with HulGen is recommended. Development and validation of a database containing HulGen parameters for existing ships would provide the user with a wider range of hulls to build from.

Further improvements to SWAD12 can be made to provide a more polished interface. Validation of SWAD12 output against SWAD90 output for a variety of SWATH hull forms is recommended. Development and validation of a database containing SWAD12 parameters for existing SWATHs would provide the user with a wider range of hull forms to use as a starting point.

These tools are all part of a larger ongoing tool development effort. Further work needs to be completed to integrate each of them into the comprehensive hydrodynamic hull form optimization toolset under development as a collaboration between George Mason University and CISD.